International Journal of Computer Vision and Image Processing
Volume 7 • Issue 4 • October-December 2017

Classifying Behaviours in Videos with Recurrent Neural Networks

Javier Abellan-Abenza, University of Alicante, Alicante, Spain
Alberto Garcia-Garcia, Department of Computer Technology, University of Alicante, Alicante, Spain
Sergiu Oprea, University of Alicante, Alicante, Spain
David Ivorra-Piqueres, University of Alicante, Alicante, Spain
Jose Garcia-Rodriguez, Department of Computer Technology, University of Alicante, Alicante, Spain

ABSTRACT

This article describes how the human activity recognition in videos is a very attractive topic among researchers due to vast possible applications. This article considers the analysis of behaviors and activities in videos obtained with low-cost RGB cameras. To do this, a system is developed where a video is input, and produces as output the possible activities happening in the video. This information could be used in many applications such as video surveillance, disabled person assistance, as a home assistant, employee monitoring, etc. The developed system makes use of the successful techniques of Deep Learning. In particular, convolutional neural networks are used to detect features in the video images, meanwhile Recurrent Neural Networks are used to analyze these features and predict the possible activity in the video.

KEYWORDS

Convolutional Neural Networks, Human Behaviour, Long-Short Term Memory, Recurrent Neural Networks, RGB-D Cameras

INTRODUCTION

Human Behavior Analysis (HBA) is a big field of interest in the artificial intelligence and computer vision community. It has many application areas like Video Surveillance, Ambient- Assisted Living, Smart Shopping Environments, etc. Supported by relevant companies in this field, the availability of human video data is growing significantly.

This work approaches HBA from the DL perspective. Deep Learning techniques have been a great step in the context of classification in the last few years due to the growth of computational power. Some of these techniques are Convolutional Neural Networks (CNNs) for image understanding, and RNNs for temporary understanding such as video or text. The main purpose of this project is to design and implement an efficient deep learning solution able to predict daily human activities recorded from RGB cameras, by using both CNNs and RNNs architectures. Moreover, an implementation of this project over Graphics Processing Units (GPUs) is aimed for comparison purposes.

DOI: 10.4018/IJCVIP.2017100101

The main goal of this work is the development of a human behavior recognition system to assist dependent people. The secondary goal is to take advantage of GPUs and accelerate the system. For this purpose, an extensive state of the art of human behavior recognition systems has been carried out. At the same time, an analysis of the available datasets has been performed in order to choose the most suitable one. For the implementation, existing deep learning frameworks have been used. To accomplish with the secondary goal, this training has been performed using GPUs to exploit parallelism. Finally, a performance analysis of our system has been performed.

After exposing the motivation and goals of this work, this paper is structured as follows: First, we present a detailed state of the art regarding HBA systems and DL techniques. Next, we present the methodology exposing the different techniques, technologies, and datasets used to carry out this work. Then, we describe the proposed solution in detail attaching the main implementation parts of our system. The experiments and discussion for each part of the implementation are detailed throughout this section. Finally, some conclusions were extracted alongside potential research directions and future works.

STATE OF THE ART

Human Behaviour Analysis (HBA) involves a wide range of applications: Video Surveillance, Ambient-Assisted Living, etc. All these applications have in common the need of creating an artificial intelligence that understands the body of a person and its natural movement for different activities. Human activities, such as "walking" or "running," are relatively easy to recognize. On the other hand, more complex activities, such as "peeling an apple," are more difficult to identify. Complex activities may be decomposed into other simpler activities, which are generally easier to recognize. Therefore, it is necessary to understand the different HBA levels that exist. Moeslund, Hilton, and Krüger (2006) defined a classification of the different action taxonomies that have been adopted later in many other works. It defines three levels of abstractions from smallest to biggest: 1. Action primitive: Basic motion recognition that represents the atomic movement out of which actions are built. 2. Action: Composed of different action primitives. 3. Activity: A higher level of abstraction which requires the semantic notion of the context and the involved objects.

Although this taxonomy is highly used among researchers, some of them use their own taxonomies, for example Ji, Liu, Li, and Brown (2008) include a higher level of abstraction called behaviour. They defined behaviour as" human motion patterns involving high-level description of actions and interactions".

Motion recognition is the fundament for detecting human activities or behaviours. Motion is decomposed in a series of poses through time. A pose can be described as the state of the body posture that can be represented by an articulated system of rigid segments connected by joints, like the model described in Andriluka, Roth, and Schiele (2009); Sapp, Toshev, and Taskar (2010).

The work of Gavrila (1999), reveals important applications of "looking at people" and reviews several lower level techniques of detecting human motion. Later, the work made by Moeslund and Granum (2001) was focused in recognizing human movements. He described a functional taxonomy of the phases required to capture human motion: Model initialization, Tracking, Pose estimation, and Recognition. This two works can be considered the early phases of human behaviour analysis. Years later, Moeslund, Hilton, and Krüger (2006), updated his work to show the latest trends in human motion capture and analysis.

HBA Recognition Systems

The human activity categorization problem has remained a challenging task in computer vision for more than two decades. Previous works on recognizing human behaviour have shown great potential with many different approaches. In Radhakrishnan et al. (2015), human activity recognition methods

are categorized into two main categories: unimodal and multi-modal according to the nature of sensor data they employ. Each of these two categories is further analyzed into sub-categories depending on how they model human activities. Figure 1 shows this classification.

Unimodal Methods

Unimodal methods represent human activities from data of a single modality, such as images, depth, skeleton, etc. Most of the existing approaches use video sequences or images to recognize human activities by extracting visual features.

Space-time approaches focus on recognizing activities based on space-time features or trajectory matching. They consider an activity in the 3D space-time concatenating 2D spaces. An activity is represented by a set of space-time features or trajectories extracted from a video sequence, H.Wang et al. (2013).

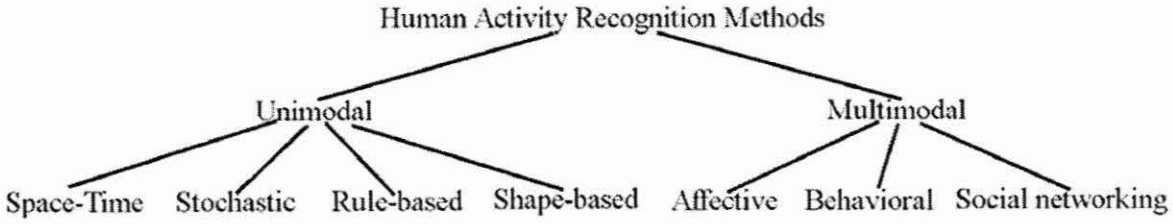
Many human activity recognition methods based on space-time representations have been proposed in the literature. Efros Mori, Efros, Berg, and Malik (2003) proposed a method which detects the optical flow of low-resolution video sequences and recognizes human actions using the nearest neighbor classifier. Schuldt, Laptev, and Caputo (2004) used space-time features to detect local events in a video, while a SVM classifier was used to recognize an action. A hierarchical approach was followed by Jhuang, Serre, Wolf, and Poggio (2007), where an input video was analyzed into several feature descriptors depending on their complexity.

In Stochastic methods, activities are considered as stochastically predictable sequences of states. Researchers have used many stochastic techniques, such as Hidden Markov Model (HMMs), Bishop (2012) and Hidden Conditional Random Fields (HCRFs), Quattoni, Wang, Morency, Collins, and Darrell (2007), to infer useful results for human activity recognition. There are representative stochastic models for action recognition such as: factorized HCRF model used by Wang and Mori Y.Wang and Mori (2009) where circle nodes correspond to variables, and square nodes correspond to factors and the hierarchical latent discriminative model proposed by Song et al. (2013).

Rule-based approaches determine ongoing events by modeling an activity using rules or sets of attributes that describe an event. Each activity is considered as a set of primitive rules/attributes, which enables the construction of a descriptive model for human activity recognition. Action recognition of complex scenes with multiple subjects was proposed by Morariu and Davis (2011). Each subject must follow a set of certain rules while performing an action. The recognition process was performed over basketball game videos, where players are detected and tracked, generating a set of trajectories which are used to create a set of spatiotemporal events. Based on the first-order logic and probabilistic approaches, such as Markov networks, the authors could infer which event has occurred.

It is well known that activity recognition algorithms based on the human silhouette play an important role in recognizing human actions. As a human silhouette consists of limbs jointly connected to each other, it is important to obtain exact human body parts from videos. This problem is considered as a part of the action recognition process. Although this part remains a challenging task, many algorithms are focused in solving this problem Lillo et al. (2014).

Figure 1. Proposed hierarchical categorization of human activity recognition methods in Radhakrishnan et al. (2015)



Multimodal Methods

Multimodal methods combine features collected from different sources. This concept can be applied to many problems: audio-visual synchronization Lichtenauer, Shen, Valstar, and Pantic (2011), tracking Perez, Vermaak, and Blake (2004) and activity recognition Wu,Wang, Deng, Chi, and Feng (2013).

Multimodal methods are based on feature fusion, which can be expressed by two different strategies: early fusion and late fusion. Early fusion directly concatenates all features into a larger feature vector and then learns the underlying action. Late fusion processes all features independently with its corresponding model and then combines all the scores with a final supervised model. In Karpathy et al. (2014) a graphical representation of this different fusion approaches is described.

Affective computing studies model the ability of a person to express, recognize, and control his/her affective states in terms of hand gestures, facial expressions, physiological changes, speech, and activity recognition Pantic and Rothkrantz (2003). This research area is generally considered to be a combination of computer vision, pattern recognition, artificial intelligence, psychology, and cognitive science. Activity recognition systems based on this method use many types of data such as electroencephalogram signals, heart and facial muscle activity, skin response, breathing, etc. Nevertheless, this type of data is more relevant for emotion and physical condition recognition systems Soleymani, Pantic, and Pun (2012), rather than for activity recognition.

Behavioural approaches aim to recognize activities (and even emotions) from multimodal data such as: video, audio, pose, gestures, facial expressions, etc. One important aspect of human behavior recognition is the choice of proper features. Metallinou and Narayanan (2013) aims to define a specific emotional state by recognizing human behaviors. A typical example of a behavior recognition system can be found in Metallinou et al. (2013) where audio-visual features and emotional annotations are fed into a Gaussian Mixture Model for estimating the emotional curves.

Social interaction can be considered as a special type of activity where someone adapts his behaviour according to the group of people surrounding him. Most of the social networking systems that affect people's behavior, such as Facebook, Twitter, and YouTube, measure social interactions and infer how such sites may be involved in issues of identity, privacy, social capital, youth culture, and education. Candamo, Shreve, Goldgof, Sapper, and Kasturi (2010) provide a complete summarization of social interactions.

Deep Learning

Deep learning is a branch of machine learning that makes use of sophisticated, multi-level "deep" neural nets to create systems that can perform feature detection from massive amounts of unlabeled training data. There is an increasing interest in the last decade for the use of deep learning due to the better performance against traditional methods such as decision trees, support vector machines, Bayesian networks, etc. The increment of computational power in the last years has made possible to consider those biologically inspired algorithms that were developed decades ago.

Convolutional Neural Networks

Convolutional Neural Networks are a special type of Artificial Neural Networks designed for processing a large amount of input data (images, audio or video). Due to the big amount of input data, feature extraction with a standard Fully Connected (FC) network would be very inefficient. What CNN does, in a broad sense, is reducing the information by looking at individual regions of the data with the purpose of retrieving relevant features. CNNs are based in filters (kernels) that behaves like the weights in the Fully Connected ANN.

The main difference with the FC weights, is that a single convolutional filter is shared along all the input regions to generate one output. This is call Local Receptive Fields, and it is very helpful to reduce the number of weights that CNN should learn. The way that the output is computed is by sliding the filter across the input. At each location, the product between each element of the kernel

and the input element it overlaps, is computed. Then all the products are summed up to obtain the output in the current location. Figure 2 represent it visually.

A common technique for reducing the number of parameters and the amount of computation in a CNN, is reducing the size of the feature maps. The pooling layer operates independently (normally after a convolutional layer) and uses the maximum or average function to reduce regions in the feature map. Combining several convolutional and pooling layers in parallel is a very good approach for detecting features. This is because different kernels size can be applied in parallel, allowing the detection of simple and complex features at different layers of the network. AlexNet (2012) was the first network that introduced that parallelism in 2012. Google Inception network is a very Deep Neural Network with high accuracy on image recognition. It combines several layers in a module called "Inception module" represented in Figure 3.

Recurrent Neural Networks

The above described methods are designed to classify independent input data, but what happens when we deal with time-series data? To tackle with these requirements, a new type of neural network was designed to model time sequence data. These networks are called Recurrent Neural Networks (RNNs) and allows the information to persist, by having loops in it (see Figure 4).

Figure 2. Convolving a 3 × 3 kernel (W) over a 4 × 4 input (X) to generate a 2 × 2 output (Y)

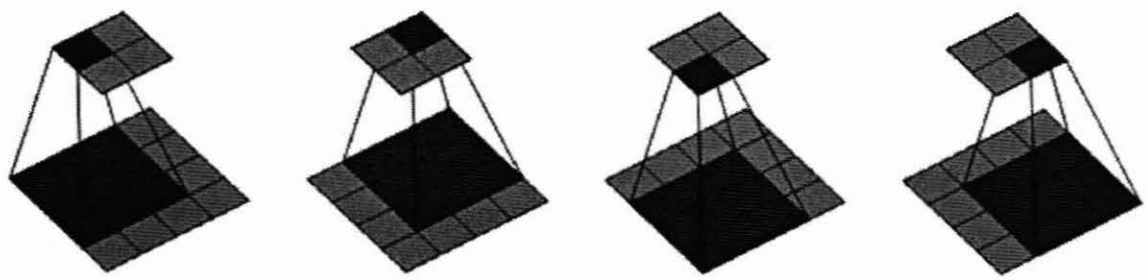


Figure 3. Inception module

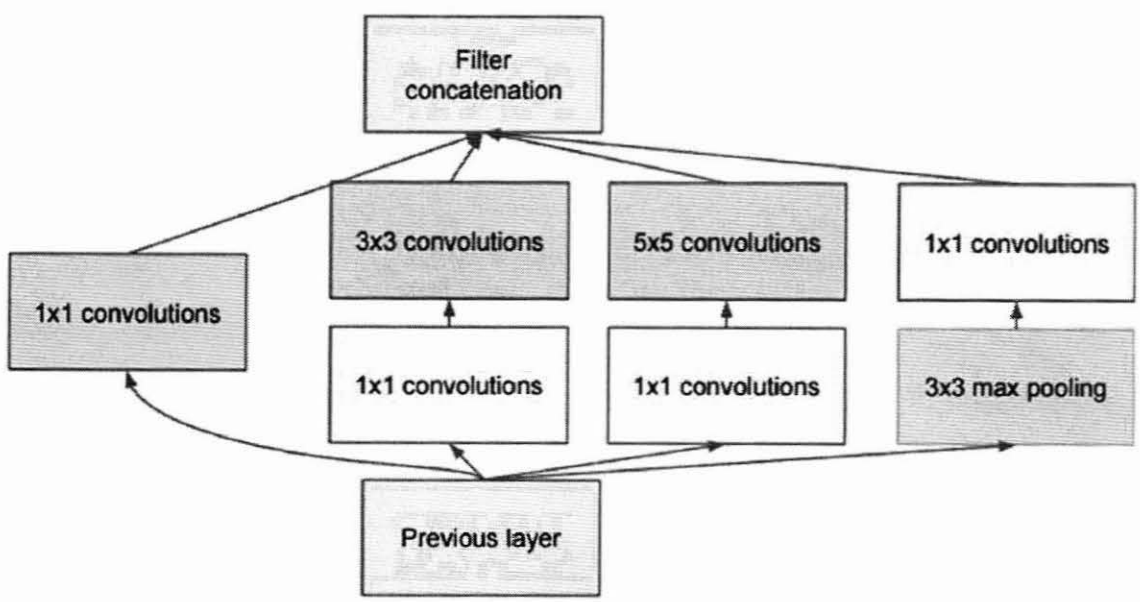
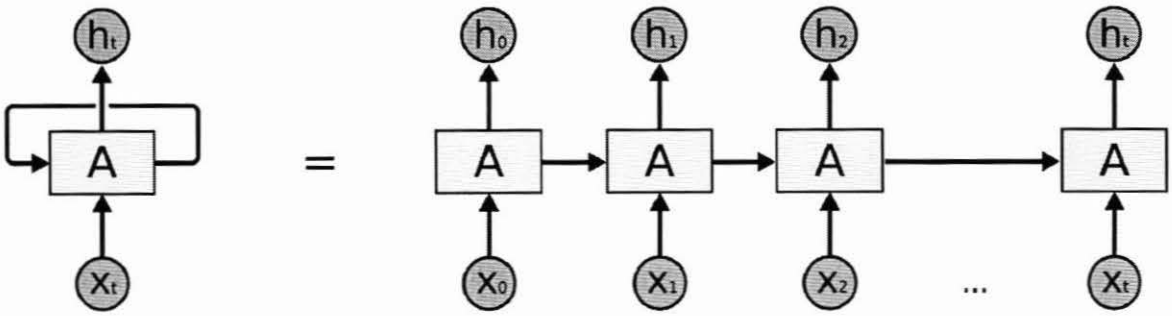


Figure 4. Recurrent neural networks have loops



In Figure 4, we see that we have a streaming input x_t and therefore a streaming output h_t. In every iteration, the output h_t will be another input for the next iteration. Loops might seem overwhelming at the beginning, but they are not such an issue if we see them in their unrolled form. Basic RNNs are useful to model small temporal dependencies. When dealing with long sequences of data (in most real cases) a new type of RNN called Long Short-Term Memory networks are used.

LSTM Networks

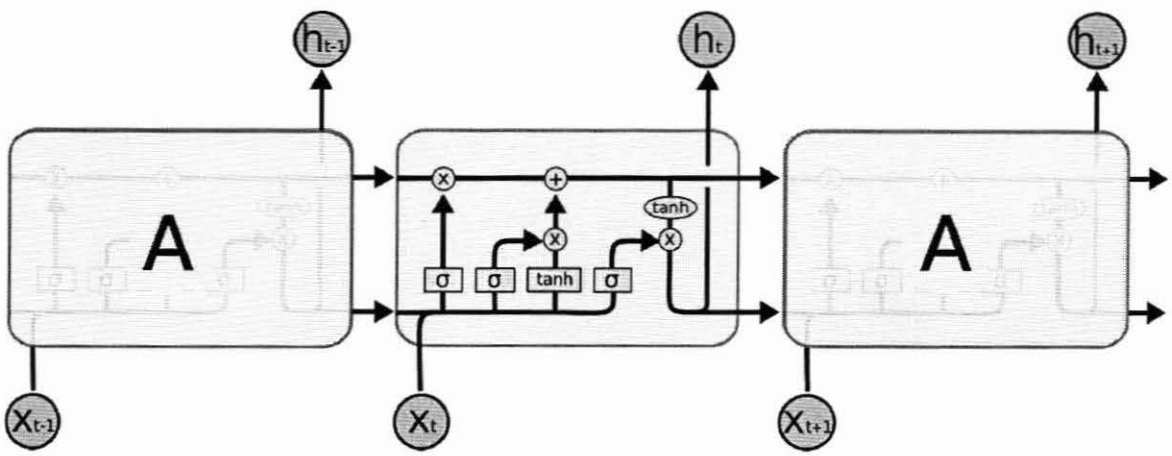
LSTM networks, introduced by Hochreiter & Schmidhuber (1997), are RNN-based models capable of learning long-term dependencies. As we noticed, vanilla RNNs have a very simple structure, such as a single perceptron with a unique tanh activation layer. In contrast, LSTM cells have a more complex structure, where instead of having a single layer (tanh), there are four, interacting in a very special way as represented in Figure 5.

METHODOLOGY

The activity recognition system will be implemented using Keras, a very popular deep learning framework. Keras is a Python wrapper that simplifies the ANN building process by offering an API that communicates with sophisticated frameworks under the hood such as Tensor Flow.

The experiments have been performed with a GPU Server equipped with an Intel(R) Core(TM) i7-5820K CPU @ 3.30GHz with DDR4 RAM memory 2666 MHz CL13 and two GPUs: NVIDIA

Figure 5. LSTM cell contains four interacting layers



GeForce Titan X 3072 CUDA cores 12 GiB of GDDR5 memory (Deep Learning) and NVIDIA Tesla K40c 2880 CUDA cores 12 GiB of GDDR5 memory (compute).

Dataset

After revising the currently available public datasets we decided to use we are UFC-101 and Activitynet datasets.

UFC101

UCF101 is an action recognition dataset of realistic action videos, collected from YouTube, having 101 action categories. This data set is an extension of UCF50 data set which has 50 action categories. With 13320 videos, UCF101 offers a high diversity of actions, variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, etc.

ActivityNet

ActivityNet dataset is a large collection of Youtube videos, labeled with the activities that appears on them. For this work, the 1.3 version of the dataset has been used. The total number of videos samples of the dataset is 19994 (849 video hours) labelled with 203 different activities. Each video is not restricted to one activity, the dataset has in average 1.41 activity categories per video.

ACTIVITY RECOGNITION SYSTEM

After reviewing the state of the art of different activity recognition systems, defining the methodology of the technologies we are going to use, and the costly task of download the two datasets, we proceed with implementing our proposal.

This system is a combination of two different DL models, a CNN reads the video frames and extract the features, and RNN reads those features and predicts the activity. This DL model will be programmed in Python, making use of the Keras framework (using the Tensorflow framework as a backend).

Once than the DL model is implemented, it is time to train with the Activitynet and UFC-101 datasets to obtain an activity recognition system. Before proceeding with this training phase, the data should be pre-processed in order to fit properly in the DL model. Finally, an analysis of the training phase will be performed. We will compare the accuracy of the different systems, and we will test them with new videos to measure their accuracy in different scenarios.

Model

The state of the art of activity recognition with deep learning indicates that the best way to approach this problem is a model with a Convolutional Neural Network, at the beginning, to extract the features of the video frames, followed by a Recurrent Neural Network able to model sequences of frames.

There are other DL models for activity recognition such as a 3D CNN which uses a FC network. In this approach, the entire video is fed at once to the 3D CNN, and this CNN is capable to extract not only image features, but also motion or time features. Then, all these features are fed to a vanilla FC network. The problem of this approach is that the whole video is needed to predict the activity. Nevertheless, with the combination of 2D CNNs and RNNs, the activity can be predicted before the video has ended, therefore this system is better because it can predict the activity in real time (early prediction).

Convolutional Part

Achieving a 2D convolutional neural network with a good performance on understanding images and generate its features (vector that summarizes the information of an image) is not an easy task. This is

because of the complexity of finding a good model and the huge amount of time and data required to train it. Due to that, a common practice in deep learning is integrating a pre-trained model to extract the features, and then, pass only the features to the new model.

There are many pretrained models for image recognition. ImageNet is a database that since 2010 organizes an annual challenge (ILSVRC) which evaluates algorithms for object detection and image classification. Due to, many DL models arise from this competition since 2012: AlexNet (2012), ZF Net (2013), VGG Net (2014), GoogLeNet (2014), Microsoft ResNet (2015).

All this deep learning models (since 2012) have in common two main blocks: the image feature extractor using convolutions, the number and arrangement of the convolutional layers determined by the model. The second block is related to the classification phase, which in this case would be a feedforward neural network taking as input a feature vector and classifying the type of object (the output size depends on the number of objects to classify). This second part is the same to every model of the ILSVRC challenge.

For this project, we are going to use the feature extraction part of a pretrained model called transfer learning which is focused on storing knowledge and applying it to a different but related problem.

The model that we are going to use is Inception v3 because has excellent classification accuracy and low computation cost. There are other models that achieve better performance such as Inception ResNet v2 but they have layers involving more computation and the classification improvement is not that much. The Inception v3 model (Figure 6) is one of the available models in the Keras framework.

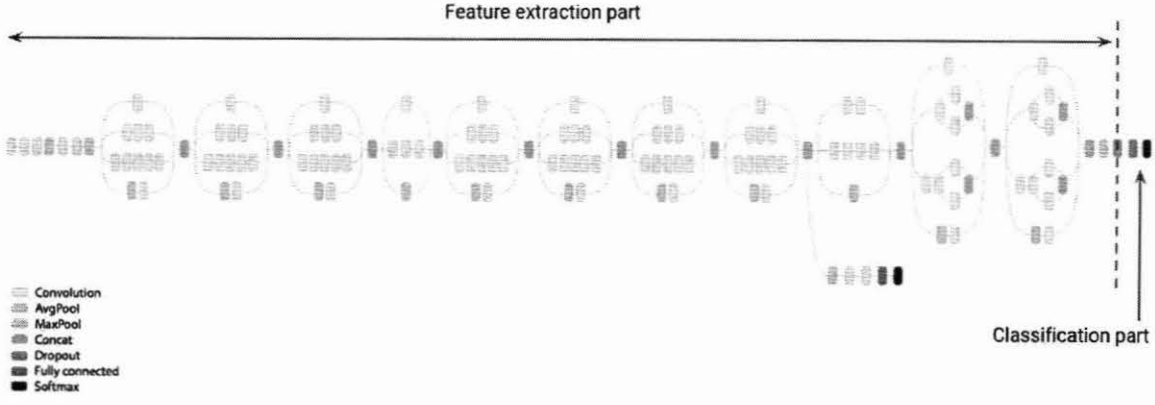
The way that inception works is the following. Instead of having a pyramid of convolutions (one behind another), Inception has, what they call inception modules, groups of layers where the flow is not sequential. In these modules, several convolutions of different size are computed separately, and then concatenated into one layer. This mechanism allows to extract more features. It also makes use of the 1x1 convolutions to reduce the number of operations.

The second part of the Inception network is the classification part, formed by fully connected layer and a softmax output layer. This classification method is suitable when we only have to classify an image at once, but if we need to classify a streaming of images, such as a video, we need RNNs.

Recurrent Part

According to the state of the art, the best method in deep learning to classify a sequence of inputs such as text, speech or video recognition, is a Recurrent Neural Network. RNNs which can model sequences of data through internal loops that feedback the network. Long Short-Term Memory networks are a type of RNNs that can "remember" important parts of the input sequence, no matter the time that have shown up (simple RNNs only remember recent parts of the sequence, they have short-term memory). In this model a LSTM network is proposed to follow the feature part of the Inception network.

Figure 6. Inception v3



The size of the feature vector that the Inception v3 network returns is 2048, so, a LSTM layer of the same size is proposed to remember every single feature of the sequence of vectors (each LSTM cell will be fed with a single feature). After the LSTM layer, a Fully Connected layer of 512 neurons is attached. Finally, a Fully Connected layer with the size of the number of activities of the dataset (101 for UFC, and 200 for ActivityNet), returns the probability for each activity.

Training

Now we have a full understanding of our dataset and our model, the next step is feed the model with the dataset in order to train it.

Extracting Features

We have seen in the previous section, that the model is composed of two different models, a CNN and a LSTM network. The CNN, whose role is obtaining the features of the frames, is already trained by Google, so only the RNN have to be trained. In order to train the LSTM network, the video data has to be converted to feature vectors. This process is called feature extraction, which feedforward the images of the videos into the CNN model (Inception V3) to obtain the feature vector of each image.

Executing the extract_features.py script, all the features are generated and stored in /data/features/. This process takes several hours depending on the machine.

Data Preparation

Once all the features have been obtained, the next step is feeding the LSTM network with those features which is not straightforward. The first logical approach is to feed the whole activity sequence (now in a feature vector format) to the LSTM network. But this is a naive approach because each activity has a different duration, and therefore, a different amount of feature vectors, and TensorFlow documentation states that is not possible to have variable length sequences, every sequence must be of the same length. So, the shape of the data has to be like:

videos × frames × features

The first approach was using the pad_sequences() function where according to the Keras documentation this function "Transform a list of sequences into sequences with a fixed length (maxlen). Sequences that are shorter than maxlen are padded with ceros at the end. Sequences longer than maxlen are truncated so that it fits the desired length."

This approach was ok, but still could be improved. Instead of padding with ceros sequences that where shorter (that does not add information), it could be repeated the sequence in a loop form until fit the desired length. And for the longer sequences, instead of retrieving just the first part of the sequence, it could be retrieving a representation of the whole sequence by skipping some frames in a organized way. This approach improves the accuracy of the system in a significant way.

Training the Model

During training, dataset is separated into training set and validation set. All the training data is forwarded to the Neural Network (NN) to train it, and the validation data is forwarded to measure the accuracy of the NN. For each training iteration or epoch, Keras shows different parameters that shows the learning improvement:

Acc: The accuracy of predicting the target class as the correct class.
Top k categorical accuracy: The accuracy of predicting the target class between the top 5 predicted class.
Loss: The loss function chosen. It will be categorical cross entropy.
Val acc: Acc for the validation set.

Val top k categorical accuracy: Top k categorical accuracy for the validation set.
Val loss: Loss acc for the validation set.

The error function is the categorical cross entropy function and represents how different two probability distributions p and q are. In order to get a good measure of the learning process, it is important to focus on the validation parameters, because these parameters are computed at the end of each epoch with the validation data, this data has not been used for training. If validation parameters getting worse, that it means the network is memorizing the training data and is not generalizing for new data, this is called overfitting. To avoid overfitting the early stopping callback is used to stop the training execution when the validation accuracy does not improve. Using dropout layers in the model is a good approach to prevent early overfitting.

The next diagrams show the val acc and val top k categorical accuracy parameters for the training phases of the UFC-101 dataset (Figure 7) and Activitynet dataset (Figure 8). Next section will analyze in detail the results obtained.

Results

This section analyzes the results obtained in the training phase for the datasets UFC-101 and Activitynet.

Evaluating the Results

Although these two datasets are very similar, there are many circumstances and factors for obtaining such difference in the accuracy (73.6% vs 62.6%). Some of these factors could be the number of samples per activity, the similarity between activities, using the same model for different number of activities to predict (100 vs 200). According to Caba Heilbron, Escorcia, Ghanem, and Carlos Niebles (2015)," ActivityNet presents more variety in terms of activity diversity and richness of taxonomy.

Figure 7. Training, accuracy on top 1 and top 5

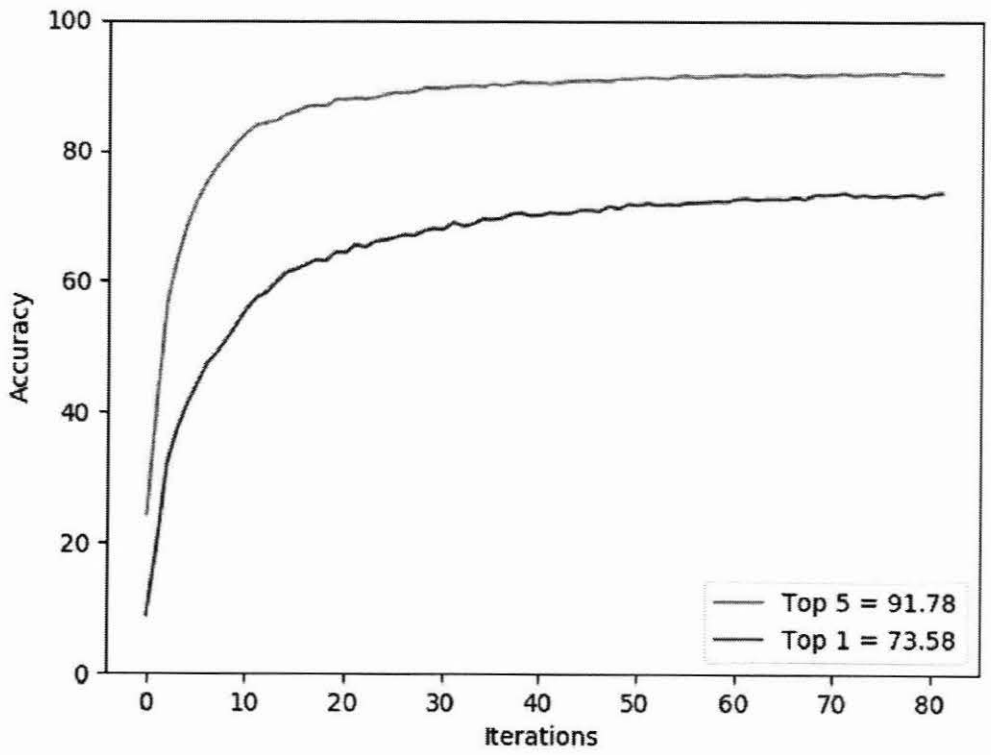
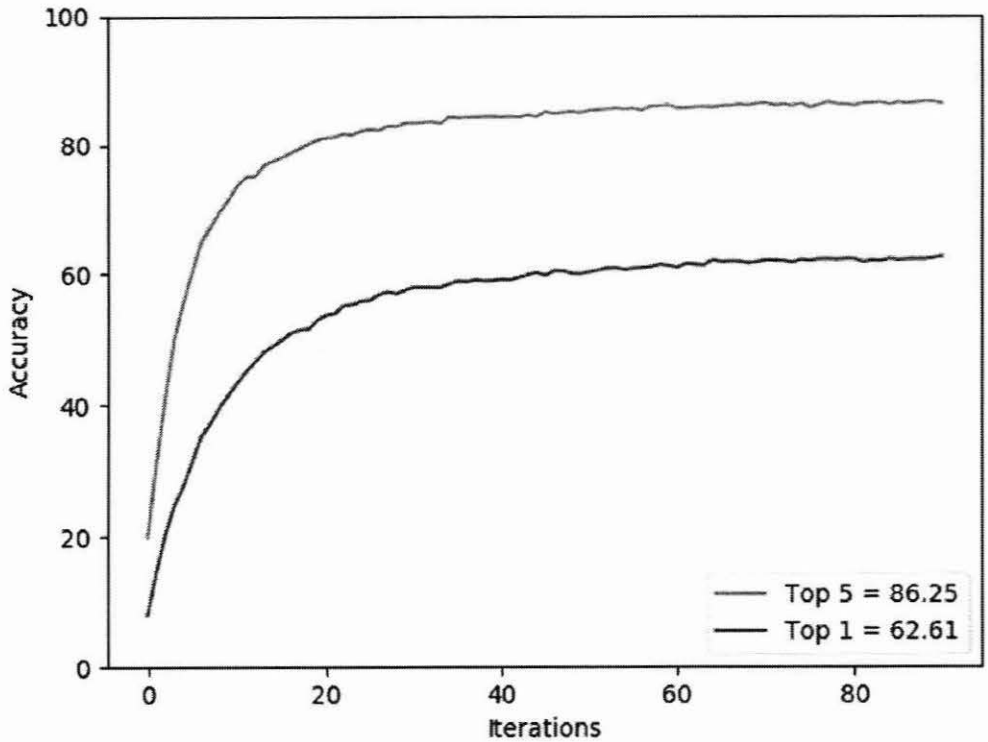


Figure 8. Activitynet: Training, accuracy on top 1 and top 5



It also contains more categories and samples per category than traditional action datasets". All these factors contribute to predict the activities in the Activinet dataset over the UFC-101.

Classification Metric

The accuracy metrics are evaluated with the total amount of videos (in the validation set) that the network succeed in the prediction. But usually, datasets have different number of videos per activity class. Therefore, a better accuracy metric could be computed. This classification metric for asserting the accuracy is the mean Average Precision (mAP). This is computed as the mean of the Average Precision (AP) of all the different classes (C) at the Dataset.

Existing Work for Activitynet

Caba Heilbron et al. (2015) shows the results obtained in the Activitynet dataset in other activity recognition systems (Table 1).

In this paper, the accuracy of activities is even lower than the achieved in this project. This is because they use other less effective techniques for this type of problem such as a combination of several types of features and a SVM as classifier.

To capture visual patterns in each input video, they construct a video representation using a combination of several feature types: motion features, static features and deep features. This is motivated by the observation that combining multiple feature types can lead to significant improvements in action recognition (Table 2):

- **Motion features:** Aiming to capture local motion patterns in a video;
- **Deep Features:** Aiming to encode information about the objects in the scene. In many activities involving object interactions, this is an important cue for disambiguation;
- **Static Features:** Aiming to encode contextual scene information. These context cues are usually helpful to discriminate human activities.

Table 1. Classification results

Category	Validation	Test
Household	34.2%	33.9%
Caring and helping	36.2%	36.7%
Personal care	41.5%	41.3%
Work-related	53.6%	53.1%
Eating and drinking	57.6%	57.2%
Socializing and leisure	63.8%	63.3%
Sports and exercises	66.6%	66.1%
Average	50.5%	50.2%

Table 2. Classification results

Feature	Untrimmed Classification	Trimmed Classification
Motion features	39.2%	47.6%
Deep features	28.7%	43.0%
Static features	24.5%	37.9%

Finally, they compare this combination of classification techniques over different datasets (Table 3).

CONCLUSION

In this work, we have implemented a human activity recognition system using a Deep Learning model. The developed system can recognize multiple activities in RGB videos by extracting features with a CNN, and finding patterns in the features with a RNN to predict the target activity. The experiments that were carried out proved the accuracy of the proposal. It has been shown that the same activity recognition system could behave differently among similar datasets.

The main highlights of this paper are: data manipulation of distinct datasets to fit a Deep Learning model; transfer learning of a pretrained Deep Learning model (Inception V3) to our system; use of LSTM Recurrent Neural Networks. The system is deployed on a server with high performance GPUs.

Table 3. Trimmed video classification

Dataset	Method	Performance
UCF101	MF, DF	85.9%
HMDB51	MF, DF	66.7%
ActivityNet	MF, DF, SF	45.9%

As future work we aim to increase the system accuracy by taken advantage of the diversity that datasets such Activitynet offers to obtain an even more robust activity recognition system. A better model could be implemented by improving the finetuning process, varying the number of layers, neurons, learning rate, etc.

ACKNOWLEDGMENT

This work has been funded by the Spanish Government TIN2016-76515-R grant for the COMBAHO project, supported with Feder funds.

REFERENCES

Andriluka, M., Roth, S., & Schiele, B. (2009). Pictorial structures revisited: People detection and articulated pose estimation. *In ieee computer society conference on computer vision and pattern recognition* (cvpr 2009).

Bishop, C. M. (2012). Pattern recognition and machine learning. Secaucus, NJ: Springer.

Caba Heilbron, F., Escorcia, V., Ghanem, B., & Carlos Niebles, J. (2015). Activitynet: A largescale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 961–970). doi:10.1109/CVPR.2015.7298698

Candamo, J., Shreve, M., Goldgof, D. B., Sapper, D. B., & Kasturi, R. (2010). Understanding transit scenes: A survey on human behavior-recognition algorithms. *IEEE Transactions on Intelligent Transportation Systems, 11*(1), 206–224. doi:10.1109/TITS.2009.2030963

Ferrari, V., Marin-Jimenez, M., & Zisserman, A. (2008). Progressive search space reduction for human pose estimation. In Proceedings of the IEEE conference on computer vision and pattern recognition.

Gavrila, D. M. (1999). The visual analysis of human movement: A survey. *Computer Vision and Image Understanding, 73*(1), 82–98. doi:10.1006/cviu.1998.0716

Jhuang, H., Serre, T., Wolf, L., & Poggio, T. (2007). A biologically inspired system for action recognition. In *Proceedings of the IEEE 11th international conference on Computer vision ICCV '07* (pp. 1–8). doi:10.1109/ICCV.2007.4408988

Ji, X., Liu, H., Li, Y., & Brown, D. (2008). Visual-based view-invariant human motion analysis: A review. In Knowledge-based intelligent information and engineering systems (pp. 741–748).

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014). Largescale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1725–1732).

Lichtenauer, J., Shen, J., Valstar, M., & Pantic, M. (2011). Cost-effective solution to synchronised audio-visual data capture using multiple sensors. *Image and Vision Computing, 29*(10), 666–680. doi:10.1016/j.imavis.2011.07.004

Lillo, I., Soto, A., & Carlos Niebles, J. (2014). Discriminative hierarchical modeling of spatiotemporally composable human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 812–819).

Metallinou, A., Katsamanis, A., & Narayanan, S. (2013). Tracking continuous emotional trends of participants during affective dyadic interactions using body language and speech information. *Image and Vision Computing, 31*(2), 137–152. doi:10.1016/j.imavis.2012.08.018

Metallinou, A., & Narayanan, S. (2013). Annotation and processing of continuous emotional attributes: Challenges and opportunities. In *Proceedings of the 2013 10th IEEE international conference and workshops on Automatic face and gesture recognition.*

Moeslund, T. B., & Granum, E. (2001). A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding, 81*(3), 231–268. doi:10.1006/cviu.2000.0897

Moeslund, T. B., Hilton, A., & Krüger, V. (2006, November). A survey of advances in vision-based human motion capture and analysis. *Comput. Vis. Image Underst.*, 104(2), 90–126. doi:10.1016/j.cviu.2006.08.002

Morariu, V. I., & Davis, L. S. (2011). Multi-agent event recognition in structured scenarios. In Proceedings of the 2011 IEEE conference on Computer vision and pattern recognition (CVPR) (pp. 3289–3296).

Mori, G., Efros, A., Berg, A., & Malik, J. (2003). Recognizing action at a distance. In Proceedings of the IEEE international conference on computer vision (ICCV '03).

Pantic, M., & Rothkrantz, L. J. (2003). Toward an affect-sensitive multimodal human-computer interaction. *Proceedings of the IEEE, 91*(9), 1370–1390. doi:10.1109/JPROC.2003.817122

Perez, P., Vermaak, J., & Blake, A. (2004). Data fusion for visual tracking with particles. *Proceedings of the IEEE, 92*(3), 495–513. doi:10.1109/JPROC.2003.823147